More Examples

on Object-Oriented Analysis and Design

Prof. T.H. Tse
Department of Computer Science
Email: thtse@cs.hku.hk
Web: hku.hk/thtse

Disclaimer

- This is not a rerun of the lectures
- but a ***recap*** of the complete analysis and design process (after learning the details at every step)
- with a view to learning ***new*** concepts as well

Recall

Find and Identify Business Classes

- Step 1. Identify actors and use cases
- Step 2. Construct a use case diagram
- Step 3. For each use case, document normal course of events
- Step 4. For each use case, document alternative courses of events
- Step 5. Identify any use case relationships
- Step 6. Find potential classes
- Step 7. Select proposed classes

Identify Actors and Use Cases

- What is a scenario?
- What is a use case?
- How many use cases?
- Is login a use case?

This is just an example

Example: Simple Bank System

User Requirements

How many use cases?

- When a customer enters an account number, the system should ask for authentication data and verify them against the bank account
- The system asks the customer to select the service type and the amount involved
- Suppose the customer selects withdrawal and enters the amount
- The system processes the transaction, dispenses the cash, and then asks whether the customer will continue
- If the customer says no, the system logs out and prints a receipt

For Each Use Case

Document Normal Course of Events

Author: T.H. Tse

Date: 01/07/2047

Use Case Name	Cash withdrawal	
Actor(s)	Customer	
Description	This use case describes the process of a customer withdrawing cash from a bank account.	
Reference	Bank-1.0	
Normal Course of Events	**Actor Actions**	**System Responses**
	Step 1. Initiate use case when a customer enters account number.	
		Step 2. Ask customer for authentication data.
	Step 3. Customer enters authentication data.	
		Step 4. Verify authentication data against the bank account.
		Step 5. Ask customer to select service type and the amount involved.
	Step 6. Customer selects withdrawal and enters amount.	
		Step 7. Process transaction.
		Step 8. Dispense cash.
	Step 9. Customer takes cash.	
		Step 10. Ask whether customer will continue.
	Step 11. Customer says no.	
		Step 12. Log out automatically.
		Step 13. Print a receipt.
	Step 14. Conclude this use case after the customer takes the receipt .	

More User Requirements

- If authentication data is invalid, ask the customer to enter again
- If account is problematic, inform the customer, do not allow the process to go on, and simply print a receipt

How many use cases?

For Each Use Case

Document Alternative Courses of Events

Alternative Courses	**Alt. Step 4A.** If authentication data is invalid, ask customer to enter again. **Alt. Step 4B.** If account is problematic, inform the customer and go to Step 13.

Pre-Conditions, Post-Conditions, and Assumptions

Alternative Courses	**Alt. Step 4A.** If authentication data is invalid, ask customer to enter again. **Alt. Step 4B.** If account is problematic, inform the customer and go to Step 13.
Pre-condition	Customer is not doing another session at the present moment.
Post-condition	Customer has logged out.
Assumptions	None at this time .

The customer logs in during the first step

Valid throughout the use case .

Useful check if there is more than one scenario

For Each Use Case
Highlight Nouns

Author: T.H. Tse Date: 01/07/2047

Use Case Name	Cash withdrawal	
Actor(s)	Customer	
Description	This use case describes the process of a customer withdrawing cash from a bank account.	
Reference	Bank-1.0	
Normal Course of Events	**Actor Actions**	**System Responses**
	Step 1. Initiate use case when a customer enters account number.	
		Step 2. Ask customer for authentication data.
	Step 3. Customer enters authentication data.	
		Step 4. Verify authentication data against the bank account.
		Step 5. Ask customer to select service type and the amount involved.
	Step 6. Customer selects withdrawal and enters amount.	
		Step 7. Process transaction.
		Step 8. Dispense cash.
	Step 9. Customer takes cash.	
		Step 10. Ask whether customer will continue.
	Step 11. Customer says no.	
		Step 12. Log out automatically.
		Step 13. Print a receipt.
	Step 14. Conclude this use case after the customer takes the receipt.	

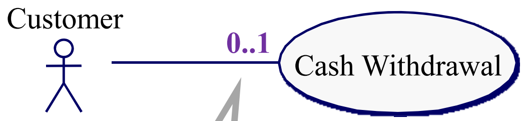
For Each Use Case
Select Proposed Classes

Author: T.H. Tse Date: 01/07/2047

Use Case Name	Cash withdrawal	
Actor(s)	Customer	
Description	This use case describes the process of a customer withdrawing cash from a bank account.	
Reference	Bank-1.0	
Normal Course of Events	**Actor Actions**	**System Responses**
	Step 1. Initiate use case when a customer enters account number.	
		Step 2. Ask customer for authentication data.
	Step 3. Customer enters authentication data.	
		Step 4. Verify authentication data against the bank account.
		Step 5. Ask customer to select service type and e amount involved.
	Step 6. Customer selects withdrawal and enters amount.	
		Step 7. Process transaction.
		Step 8. Dispense cash.
	Step 9. Customer takes cash.	
		Step 10.
	Step 11. Customer says no.	
		Step 12.
		Step 13.
	Step 14. Conclude this use case after the customer takes the receipt.	

Only 1 proposed class for this example .

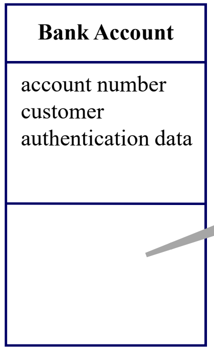
Use Case Relationship



More marks if we add <<include>> and <<extend>>?

Less marks if not used properly .

Analysis Class Diagram



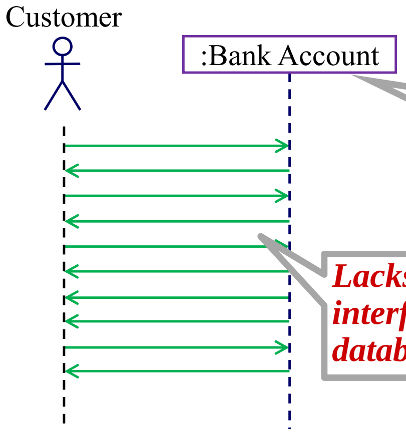
Bank Account

account number
customer
authentication data

Must be empty

Why? .

Sequence Diagram



Customer

:Bank Account

Traditional system with one class

Lacks flexibility: user interface, control, and database are hardcoded .

Recall

Object-Oriented *Design* Process

- Realize the use case model to reflect the implementation environment
- Model object interactions and behaviour that support the use case scenario
- Update the class diagram to reflect the implementation environment .

Example

Before Design

Step 2. Ask the customer for authentication data .

Example
After Design

Step 2. Display a ***dialogue window*** requesting for Password.

Step 3. Customer enters Password and ***clicks 〈OK〉***.

Step 4. If Password is ***not numeric***, ***prompt*** customer to enter again.

Step 5. Otherwise, check Password against Bank Account.

Step 6. If Password is ***invalid***, ask the customer for an alternative means of authentication through a ***dropdown list***, which includes ***the use of security device, naming the favourite musician, naming mother's maiden name, verifying finger print, and verifying iris of the eyes***.

Step 7. Customer ***clicks*** an item on the dropdown list.

Step 8. If the customer chooses ...

Analysis Use Case
Document Alternative Courses of Events

Alternative Courses	**Alt. Step 4A.** If authentication data is invalid, ask customer to enter again. **Alt. Step 4B.** If account is problematic, inform the customer and go to Step 13.
Pre-condition	Customer is not doing another session at the present moment.
Post-condition	Customer has logged out.
Assumptions	None at this time.

Design Use Case
Document Alternative Courses of Events

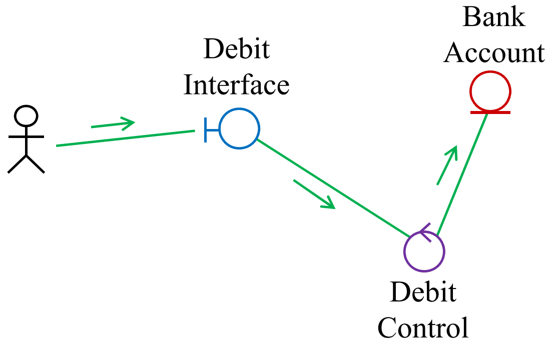
Alternative Courses	**Alt. Step 4A.** If ***Password*** is invalid, ask Customer to enter again. **Alt. Step 4B.** If Account is problematic, ***display Error Window*** and go to Step 13. **Alt. Step X. ...** **Alt. Step Y. ...** **Alt. Step Z. ...**
Pre-condition	Customer is not doing another session at the present moment.
Post-condition	Customer has logged out.
Assumptions	None at this time.

Add more design alternatives

Design-based now

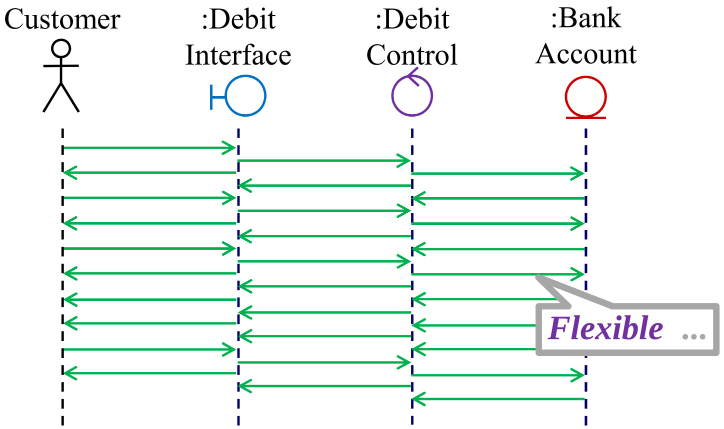
Design alternatives:
- Disallow the new session?
- Kill the existing session? .

Model High-Level Object Interactions for a Use Case
Communication Diagram



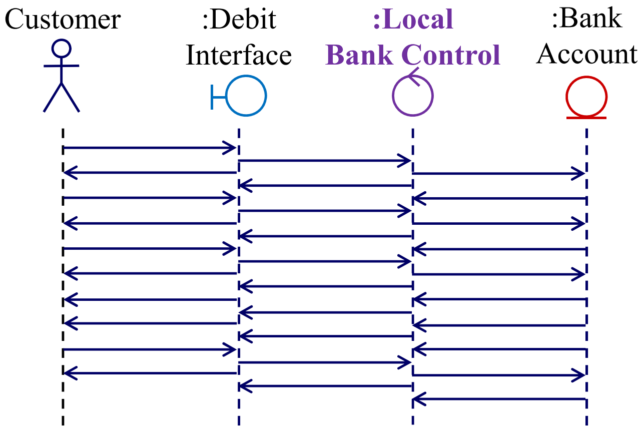
Model Detailed Object Interactions for a Use Case

Sequence Diagram



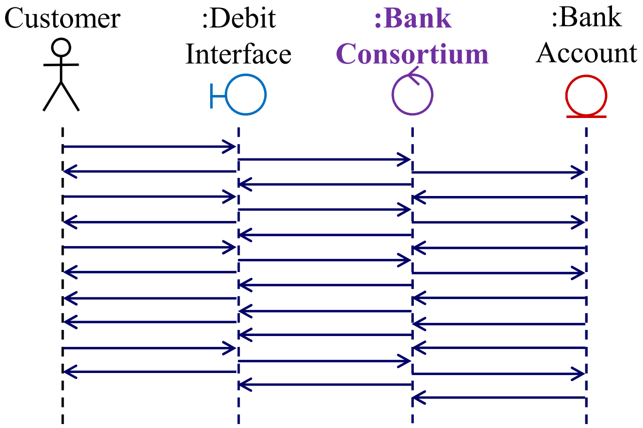
Model Detailed Object Interactions for a Use Case

Sequence Diagram



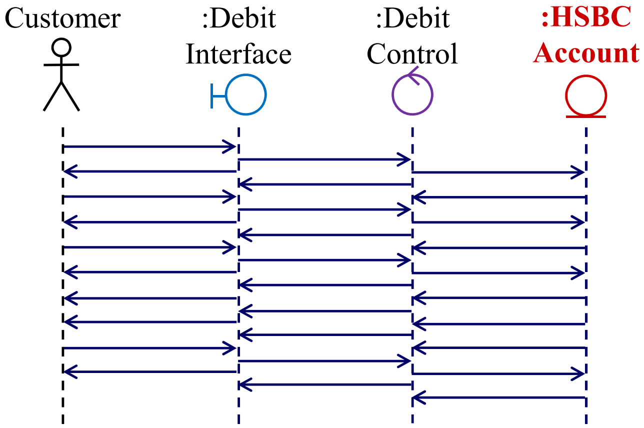
Model Detailed Object Interactions for a Use Case

Sequence Diagram



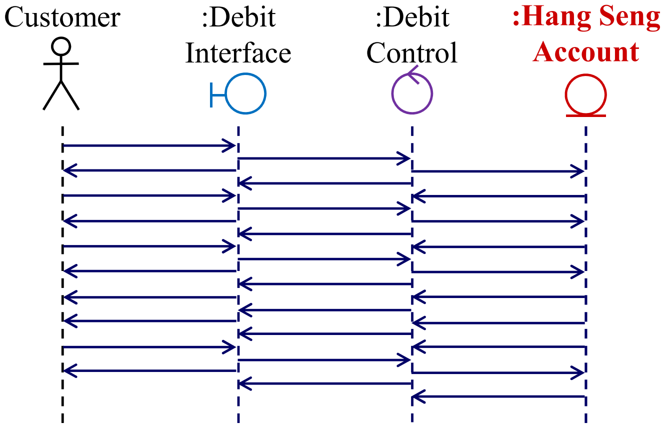
Model Detailed Object Interactions for a Use Case

Sequence Diagram



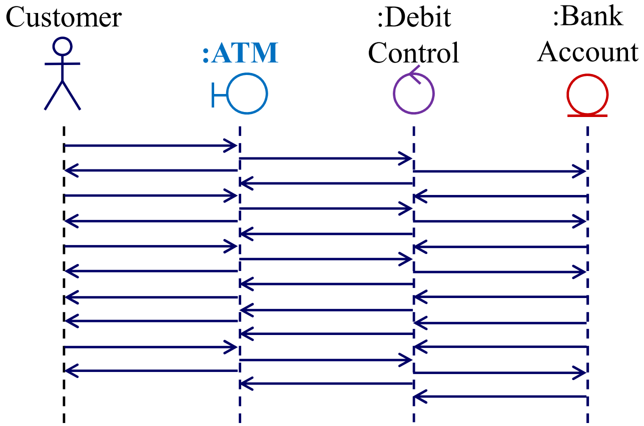
Model Detailed Object Interactions for a Use Case

Sequence Diagram



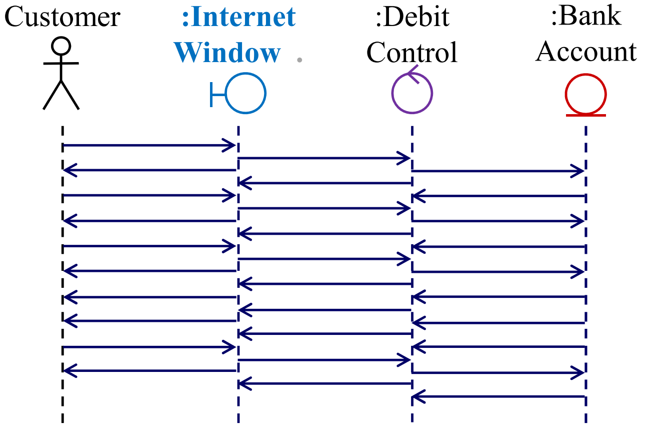
Model Detailed Object Interactions for a Use Case

Sequence Diagram



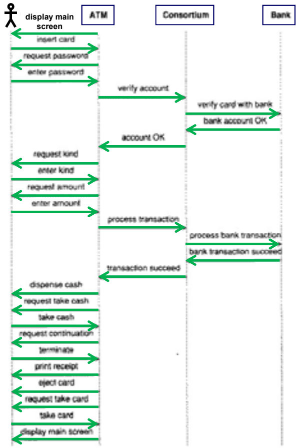
Model Detailed Object Interactions for a Use Case

Sequence Diagram

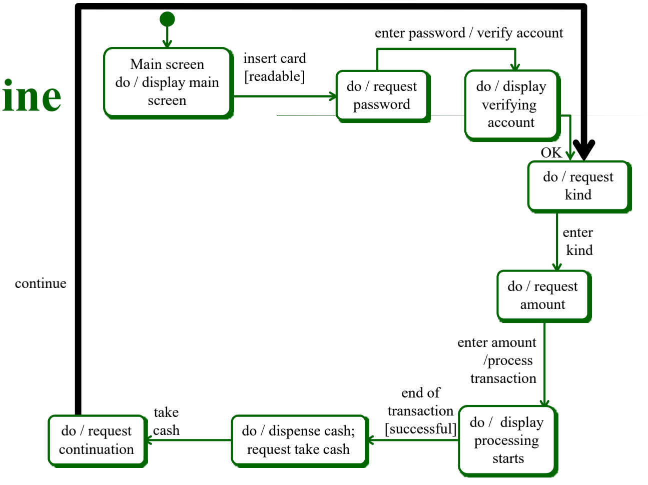


Model Detailed Object Interactions for a Use Case

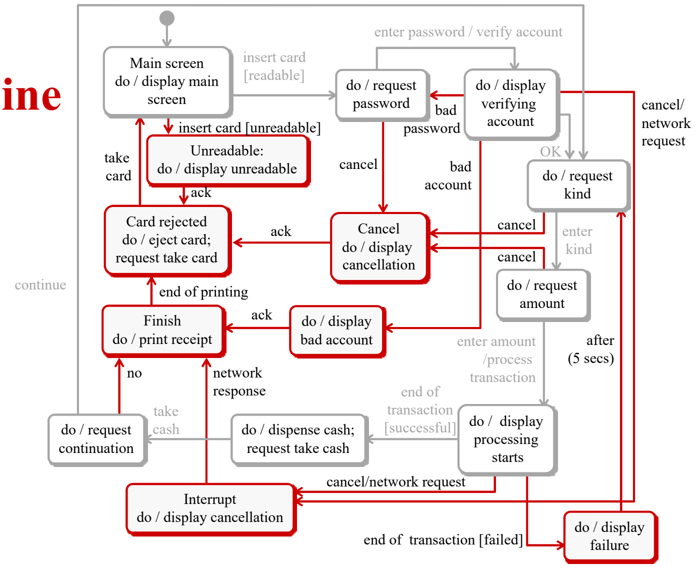
Sequence Diagram



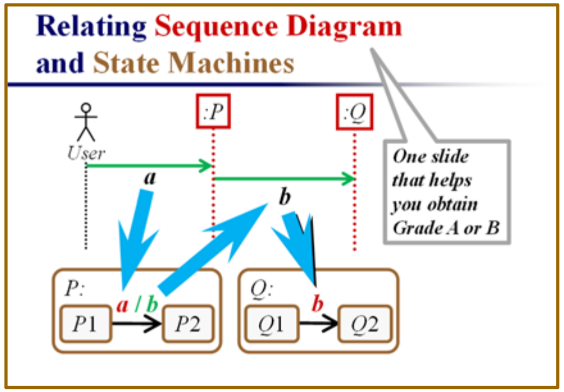
State Machine for Class ATM


Main screen
do / display main screen
insert card [readable]
do / request password
enter password / verify account
do / display verifying account
OK
do / request kind
enter kind
do / request amount
enter amount /process transaction
do / display processing starts
end of transaction [successful]
do / dispense cash; request take cash
take cash
do / request continuation
continue

State Machine for Class ATM



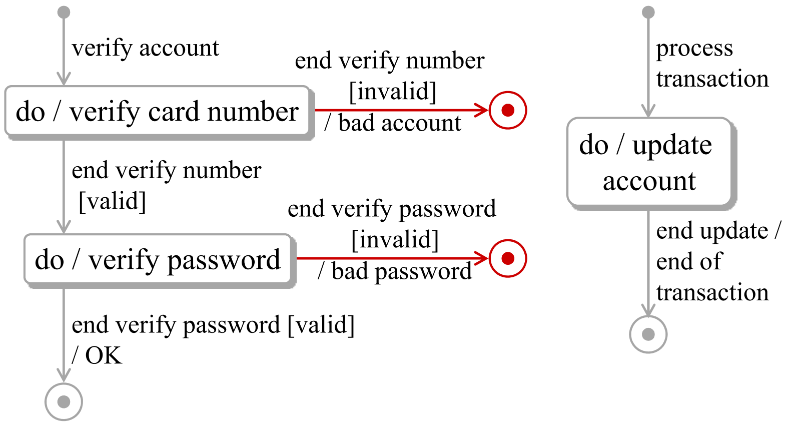
Recall



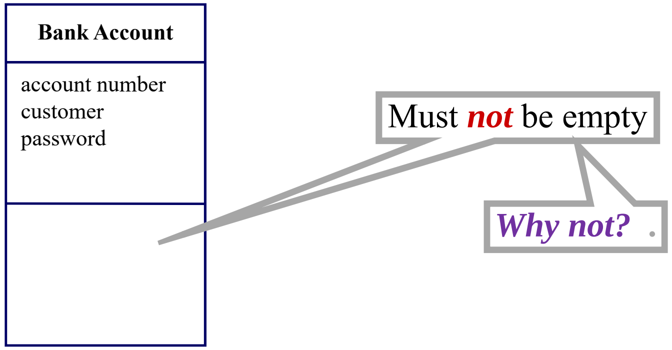
Relating Sequence Diagram and State Machines

One slide that helps you obtain Grade A or B

State Machine for Other Class(es)


verify account
do / verify card number
end verify number [invalid] / bad account
end verify number [valid]
do / verify password
end verify password [invalid] / bad password
end verify password [valid] / OK
process transaction
do / update account
end update / end of transaction

Design Class Diagram



Must ***not*** be empty

Why not?

Use Cases, Class Diagrams, Sequence Diagrams, and State Machines

- How many use cases?
- How many class diagrams?
- How many sequence diagrams?
- How many state machines?